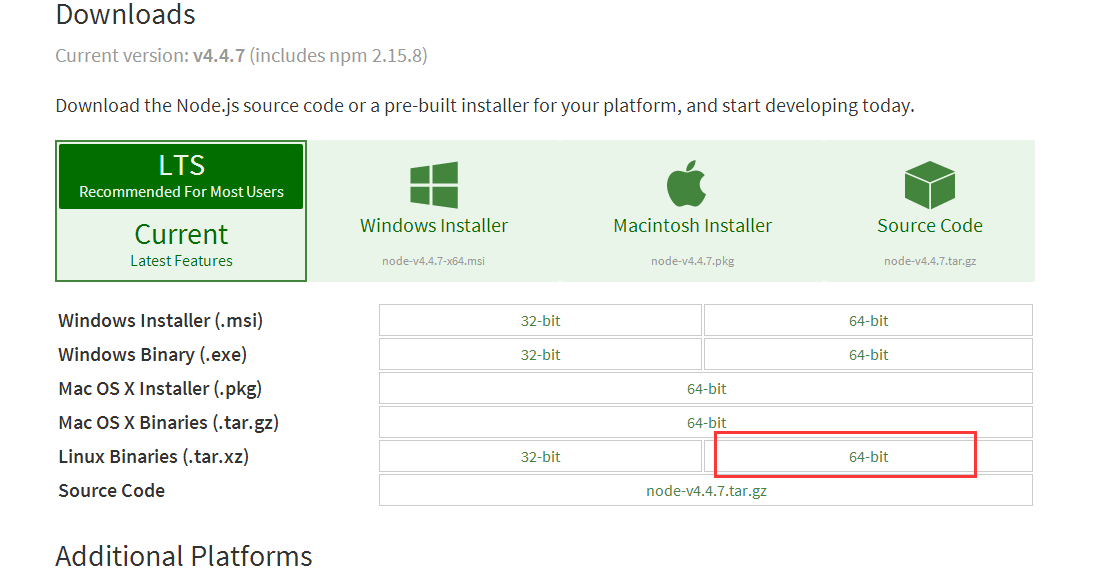
操作系统： CentOS 6.5 64位

**【0】下载**

<https://nodejs.org/en/download/>

nodejs的官网，



我下的是64位。

文件的上传：上传到服务器的话，我是用ftp，对于我这种新手来说，用ftp来处理文件的转移/复制/粘贴最方便了。

如果是虚拟机的话，我不太清楚，反正感觉linux下载文件挺麻烦的。。。

不过或许可以用图形化的linux操作系统，然后用浏览器下载？

**【1】.tar.xz解压缩方法**

### xz压缩文件方法或命令

* xz -z 要压缩的文件
* 如果要保留被压缩的文件加上参数 -k ，如果要设置压缩率加入参数 -0 到 -9调节压缩率。如果不设置，默认压缩等级是6.

### xz解压文件方法或命令

* xz -d 要解压的文件
* 同样使用 -k 参数来保留被解压缩的文件。

### 创建或解压tar.xz文件的方法

* 习惯了 tar czvf 或 tar xzvf 的人可能碰到 tar.xz也会想用单一命令搞定解压或压缩。其实不行 tar里面没有征对xz格式的参数比如 z是针对 gzip，j是针对 bzip2。
* 创建tar.xz文件：只要先 tar cvf xxx.tar xxx/ 这样创建xxx.tar文件先，然后使用 xz -z xxx.tar 来将 xxx.tar压缩成为 xxx.tar.xz
* **解压tar.xz文件：先 xz -d xxx.tar.xz 将 xxx.tar.xz解压成 xxx.tar 然后，再用 tar xvf xxx.tar来解包。**

**【2】.tar解压缩方法**

**tar xvf filename.tar**

**③判断自己下载的文件，然后编译**

如果跟我下的是一样，那么下载的是编译好的文件，

如何判断？

简单说就是解压后，在bin文件夹中已经存在node以及npm，如果你进入到对应文件的中执行命令行一点问题都没有，不过不是全局的，所以将这个设置为全局就好了。

cd node-v0.10.28-linux-x64/bin

ls

./node -v

**注：**

①第一行命令的node-v0.10.28-linux-x64就是你看到的nodejs解压缩后的那个文件夹（可能有所不同）例如我的文件夹名字是node-v4.4.7-linux-x64。

②第三行命令是查看nodejs的版本，如果能成功查看说明没问题

这就妥妥的了，node文件夹具体放在哪，叫什么名字随你怎么定。

例如我的nodejs解压缩后的文件夹是/usr/nodejs4.4.7/node-v4.4.7-linux-x64

然后设置全局：

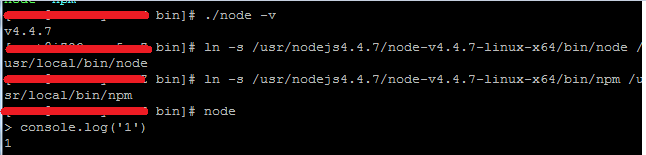
ln -s /usr/nodejs4.4.7/node-v4.4.7-linux-x64/bin/node /usr/local/bin/node

ln -s /usr/nodejs4.4.7/node-v4.4.7-linux-x64/bin/npm /usr/local/bin/npm

这里/usr/nodejs4.4.7/node-v4.4.7-linux-x64这个路径是你自己放的，你将node文件解压到哪里就是哪里。（也就是你在上面看到的node文件和npm文件夹）

**注：**

①如果正常的话，应该是下图这样，可以通过输入node来成功运行代码的



②如果不是编译好的版本，我给个参考链接：

<http://www.xitongzhijia.net/xtjc/20150202/36680.html>

**以下似乎是处理未编译的源代码的方法**

**【2.1】CentOS更新yum**

yum update

**【2.2】安装g++**

如果你使用 Linux，那么你需要使用g++ 来编译 Node.js。

在**Debian/Ubuntu**中，你可以通过 apt-get install g++ 命令安装g++。

在**Fedora/Redhat/CentOS**中，你可以使用 yum install gcc-c++ 安装。

**【2.3】另一个工具**

libssl-dev 是调用 OpenSSL 编译所需的头文件，用于提供 SSL/TLS 加密支持。Mac OSX 的 Xcode 内置了 libssl-dev。

在 Debian/Ubuntu 中，你可以通过 apt-get install libssl-dev 命令安装。

在 Fedora/Redhat/CentOS 中， 你可以通过 yum install openssl-devel 命令安装。

同样，你也可以访问 http://openssl.org/ 下载一个。

【2.4】

接下来，进入 Node.js 源代码所在目录，运行：

./configure

make

sudo make install

18 第2 章 安装和配置Node.js

之后大约等待20分钟，Node.js 就安装完成了，而且附带安装了 npm。

如果你使用 Mac OS X，还可以尝试使用 homebrew 编译安装 Node.js。

首先在 http://mxcl.github.com/homebrew/获取 homebrew，然后通过以下命令即可自动解析编译依赖并安装Node.js：

brew install node

【2.5】测试下能否访问：

输出node，进入nodejs的命令行模式，

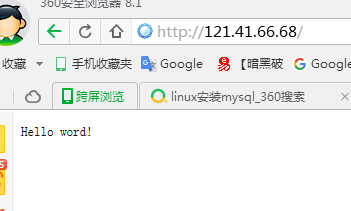
然后复制以下代码到linux（可以直接全部复制粘贴）；

var http = require("http");  
var url = require("url");  
function onRequest(request, response) {  
 var pathname = url.parse(request.url).pathname;  
 console.log("Request for " + pathname + " recived.");  
 response.writeHead(200, {"Content-type": "text/plain"});  
 response.write("Hello word!");  
 response.end();  
}  
http.createServer(onRequest).listen(80);  
console.log("Server has started!");

如果是本机的话，访问127.0.0.1；

如果是服务器的话，访问服务器的ip地址

正常的话，你会看到如下页面：



**退出：**

按Ctrl+d可以退出nodejs，退出后，页面即无法访问了

**【3】安装mysql**

安装攻略（可供参考）

<http://jingyan.baidu.com/album/a378c9609eb652b3282830fd.html?picindex=2>

我下载的是**mysql-5.7.13-linux-glibc2.5-x86\_64.tar.gz**

因此首先要解压缩：

①进入到该文件在linux的目录；

②输入以下命令解压缩：

tar -xvf mysql-5.7.13-linux-glibc2.5-x86\_64.tar.gz

注意，输入文件名的时候，可以尝试使用tab直接补全（有的是支持的）

附解压缩相关说明：

* 总结  
  1、\*.tar 用 tar –xvf 解压  
  2、\*.gz 用 gzip -d或者gunzip 解压  
  3、\*.tar.gz和\*.tgz 用 tar –xzf 解压  
  4、\*.bz2 用 bzip2 -d或者用bunzip2 解压  
  5、\*.tar.bz2用tar –xjf 解压  
  6、\*.Z 用 uncompress 解压  
  7、\*.tar.Z 用tar –xZf 解压  
  8、\*.rar 用 unrar e解压  
  9、\*.zip 用 unzip 解压
* 来源：<http://blog.csdn.net/mmllkkjj/article/details/6768294>

③解压缩完成后，复制mysql目录，输入：

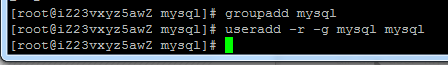
cp mysql-5.7.13-linux-glibc2.5-x86\_64 /usr/local/mysql -r

这行命令是将mysql目录复制到系统的本地软件目录，复制过程**无提示**，请耐心等待复制结束。

文件挺多挺大，要复制蛮久的

* 附：
* linux的cp命令（复制）
* <http://www.cnblogs.com/xd502djj/archive/2011/11/25/2263562.html>

④添加系统mysql组和mysql用户：



**注：原文链接的图中，第一步好像有反应，但我的没有，不太明白**

⑤安装数据库：

* 首先进入local下的mysql目录

cd /usr/local/mysql

* 添加系统mysql组和mysql用户：

执行命令：groupadd mysql和useradd -r -g mysql mysql

* 修改当前目录拥有者为mysql用户：

chown -R mysql:mysql ./

* 创建data目录：

mkdir data

* 修改文件：

vi /etc/my.cnf

按i进入修改模式，修改为以下（**应该是这样吧？求指正，主要是红色部分**）

[mysqld]

basedir=/usr/local/mysql/

datadir=/usr/local/mysql/data

socket=/var/lib/mysql/mysql.sock

user=mysql

# Disabling symbolic-links is recommended to prevent assorted security risks

symbolic-links=0

[mysqld\_safe]

log-error=/var/log/mysqld.log

pid-file=/var/run/mysqld/mysqld.pid

* 进入mysql安装目录:

cd /usr/local/mysql

=====================分割线开始============================

**注：以下代码可能出现问题，建议先看到分割线结束，使用分割线结束后的流程。**

* 执行（这是一整行，建议复制粘贴）初始化数据：

bin/mysqld --user=mysql --basedir=/usr/local/mysql/ --datadir=/usr/local/mysql/data/ --secure-file-priv=/root/ --initialize

* 注意: 在初始化时如果加上 --initial-insecure，则会创建空密码的 root@localhost 账号 --secure-file-priv 指定密码文件夹位置 对应文件夹下就会有 .mysql\_secret 文件（5.6以前） 之后的会在log中提示 A temporary password is generated for root@localhost: (U?0T,6,qQ2: (U?0T,6,qQ2: 这个就是第一次用来登陆mysql的密码一定要记住
* 添加开机启动

cp support-files/mysql.server /etc/init.d/mysql

* 启动mysql服务

service mysql start

发现启动失败：

查看执行那一步的报错（第一行），提示

[Warning] TIMESTAMP with implicit DEFAULT value is deprecated. Please use --explicit\_defaults\_for\_timestamp server option (see doc umentation for more details).

输入：

vi /etc/my.conf

找到“[mysqld]”设置项，添加代码：

#开启查询缓存

explicit\_defaults\_for\_timestamp=true

然后继续输入上面【执行】那一行，发现报错：

Insecure configuration for --secure-file-priv: Current value does not restrict location of generated files. Consider setting it to a valid, non-empty path.

搜索解决方案，发现原因是

官方文档说了mysqld --initialize-insecure自动生成无密码的root用户，mysqld --initialize自动生成带随机密码的root用户。data文件夹不为空是不能执行这个命令的。可以先删除data目录下的所有文件或者移走。

（**注：**此时发现在上面那一长串命令时，--secure-file-priv前两个“-”和后面之间多个空格，可能是因为多一个空格的原因，不确定是哪个的问题导致这个报错，故全列出来做参考。另，源代码中多出的空格我已经修改掉了）

删除mysql下的data文件夹（记住，是文件夹，包含里面所有文件），这次改输入

bin/mysqld --user=mysql --basedir=/usr/local/mysql/ --datadir=/usr/local/mysql/data/ --secure-file-priv=/root/ --initialize-insecure

这种方式是创建空密码的 root@localhost

输入时提示：

[Warning] InnoDB: New log files created, LSN=45790

2016-07-23T16:20:43.189767Z 0 [Warning] InnoDB: Creating foreign key constraint system tables.

2016-07-23T16:20:43.283044Z 0 [Warning] No existing UUID has been found, so we assume that this is the first time that this server has been started. Generating a new UUID: 675c6f7c-50f1-11e6-aaa1-00163e006ada.

2016-07-23T16:20:43.287196Z 0 [Warning] Gtid table is not ready to be used. Table 'mysql.gtid\_executed' cannot be opened.

2016-07-23T16:20:43.288147Z 1 [Warning] root@localhost is created with an empty password ! Please consider switching off the --initialize-insecure option.

=====================分割线结束============================

[mysqld]

basedir=/usr/local/mysql

datadir=/usr/local/mysql/data

#socket=/var/lib/mysql/mysql.sock

socket=/tmp/mysql.sock

user=mysql

# Disabling symbolic-links is recommended to prevent assorted security risks

symbolic-links=0

[mysqld\_safe]

log-error=/var/log/mysqld.log

pid-file=/var/run/mysqld/mysqld.pid

~

~

~

**命令集合：**

* linux查看进程 ps 命令详解
* <http://blog.sina.com.cn/s/blog_79870b4701013f4v.html>

**【4】进程的查看和终止**

ps 命令用于查看当前正在运行的进程。

　　grep 是搜索

　　例如： ps -ef | grep java

　　表示查看所有进程里CMD是java的进程信息

　　ps -aux | grep java

　　-aux 显示所有状态

　3. kill 命令用于终止进程

　　例如： kill -9

**【6】把启动的app.js放在后台运行：**

原本使用npm start的地方，输入

nohup npm start&

即可

**【7】启动mysql**

/etc/init.d/mysql start

<http://blog.csdn.net/wy3552128/article/details/8143686>

**【8】高版本5.7以上修改初始化密码**

set password=password('123456');

源链接：

<http://bbs.bestsdk.com/detail/762.html>

**【9】删除**

删除命令是rm

删除文件是：rm 文件名

不需要确认的删除文件是：rm -f 文件名

删除目录是（目录和他的所有子文件）：rm -r 目录名

不需要确认的删除目录：rm -rf 目录名

**【10】用yum安装mysql**

先看这个

<http://www.2cto.com/database/201207/141878.html>

让mysql开机自动启动看

<http://www.cnblogs.com/Anker/p/3551508.html>

**chkconfig --level 2345 mysqld on**

**【11】mysql允许某个ip访问**

<http://www.360doc.com/content/16/0101/23/3969253_524712845.shtml>

mysql> use mysql

mysql> GRANT ALL ON \*.\* to root@'113.140.251.163' IDENTIFIED BY 'wangdong1229wangdong';

mysql> FLUSH PRIVILEGES;

设置：

允许查

GRANT select ON job.account TO 'jobUser'@'%' IDENTIFIED BY 'jobUserPassword';

允许改某表的某些字段：

GRANT update (lastVisitTime, lastVisitIp) ON job.account TO 'jobUser'@'%' IDENTIFIED BY 'jobUserPassword';

允许在某表上增查改：

GRANT insert,select,update ON job.errorlog TO 'jobUser'@'%' IDENTIFIED BY 'jobUserPassword';

GRANT insert,select,update ON job.joblist TO 'jobUser'@'%' IDENTIFIED BY 'jobUserPassword';

GRANT insert,select,update ON job.visitsrecord TO 'jobUser'@'%' IDENTIFIED BY 'jobUserPassword';

**【12】查询ip**

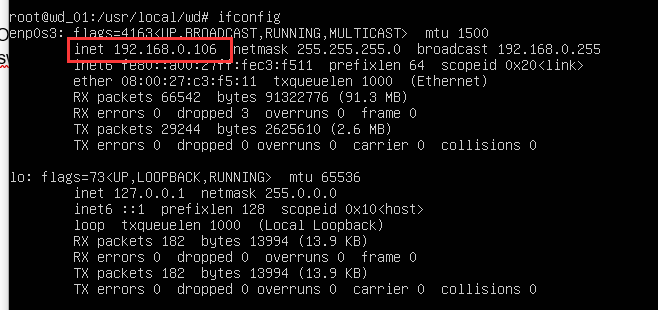
命令：ifconfig

如果不行的话，就使用命令：ip addr

第二种如果不显示ip地址，参考这个文章：

<https://blog.csdn.net/dancheren/article/details/73611878>

Ip地址如下图红框



**【13】查看linux系统的版本号**

命令： cat /proc/version



**【14】Shell脚本**

说明链接：

<https://github.com/qinjx/30min_guides/blob/master/shell.md>

简单来说：

* #!/bin/sh 固定为第一行
* 部分系统需要：每行一个命令，除了第一行和最后一行外，其他每行加 && 结尾（例如：cd /abc &&）
* 运行的时候，执行 ./test.sh ，其中test.sh 是sh文件名

**【15】下载**

<http://www.cnblogs.com/peida/archive/2013/03/18/2965369.html>

wget [参数] [URL地址]

**【16】访问网页**

curl 网页链接

返回内容是网页的html代码

**【17】shell脚本中执行时提示“没有那个文件或目录”**

问题分析：  
1、将windows 下编写好的SHELL文件，传到linux下执行，提示出错。  
2、出错信息：bad interpreter: 没有那个文件或目录。

问题原因：  
因为操作系统是windows，在windows下编辑的脚本，所以有可能有不可见字符。脚本文件是DOS格式的  
即每一行的行尾以\r\n来标识, 其ASCII码分别是0x0D, 0x0A.  
  
解决方法：  
可以有很多种办法看这个文件是DOS格式的还是UNIX格式的, 还是MAC格式的  
（1） vim filename  
**然后用命令 :set ff**  
可看到dos或unix的字样，如果的确是dos格式的, 那么用:set ff=unix把它强制为unix格式的,，然后存盘退出后就可运行。

**【18】查看端口是否开放**

telnet ip port *# telnet 59.110.155.32 8001*

如果没有 telnet命令，则 yum install telnet 安装

**【19】netstat命令不可用**

执行yum install net-tools 即可

**【20】Shell脚本实现对文件编辑**

<https://blog.csdn.net/qq_37674858/article/details/80066264>

**【21】设置环境变量**

简单来说，原本需要在/usr/local/aaa/ 下执行 abc，才能运行 /usr/local/aaa/abc 这个文件。

我现在想在任何地方执行 abc 都能自动执行 /usr/local/aaa/abc ，就需要配置环境变量。

所谓环境变量，简单来说，就是让你在任何一个地方，都可以通过文件名直接执行某个原本可能目录很深的文件。

环境变量配置在 /usr/bin 目录。

所以就很简单了，软连接到该目录里即可。例如：ln -s /usr/local/nginx/sbin/nginx /usr/bin/ ，那么执行 nginx的时候，就相当于执行 /usr/local/nginx/sbin/nginx

**【22】软连接**

功能介绍：

* 假如有2个文件夹A和B，a1文件在文件夹A中；
* 想在B文件夹中，直接访问/修改a1文件；
* 也可以继续在A文件夹中访问a1文件；
* 相当于windows的快捷方式；
* 也类似A和B都有一个a1文件的指针（不严格准确）；
* 修改A或者B里面的a1，都会让另外一个文件夹里的a1同步修改；

命令（软链接）：

ln -s 被链接的文件的路径(a1文件) 目标目录

示例：

ln -s /usr/A/a.html /Users/mac/Desktop/B/

其他：

* 以上是软链接，A文件夹里的a1是源文件，B里面的a1是一个引用；
* B里面的a1删除，不影响A里面的a1；
* A里面的a1删除，B里面的a1会失效；
* 除此之外，还有硬链接，A或B里面任意一个a1删除，都不影响另外一个，并且他们之间内容变更共享；
* 也支持目录的链接；

**【23】查找文件**

命令：

find 文件路径 参数

示例：

// 在根目录下，查找 nginx.pid 文件

find / -iname "nginx.pid"

MAC上比较好用的搜索：

mdfind -name 文件名

示例：

mdfind -name "nginx.pid"

// 输出 /usr/local/var/run/nginx.pid

**【24】后退到上一个目录**

命令

cd -

说明：

* 就像window里的后退按钮那样；
* 但只有一步缓存，即A文件夹到B文件夹后，输入这个命令后跳回A，再输入又跳回B；

**【25】移动文件**

命令：

mv [选项] 源文件或目录 目标文件或目录

示例：

移动前文件树：

|-- a （文件夹）

| |-- 1.md

|-- b （文件夹）

命令行定位在 b 文件夹下，执行命令：

mv ../a/1.md ./

移动后文件树：

|-- a （文件夹）

|-- b （文件夹）

|-- 1.md

参数：

-t ： --target-directory=DIRECTORY move all SOURCE arguments into DIRECTORY，即指定mv的目标目录，该选项适用于移动多个源文件到一个目录的情况，此时目标目录在前，源文件在后。

-b ：若需覆盖文件，则覆盖前先行备份。

-f ：force 强制的意思，如果目标文件已经存在，不会询问而直接覆盖；

-i ：若目标文件 (destination) 已经存在时，就会询问是否覆盖！

-u ：若目标文件已经存在，且 source 比较新，才会更新(update)

参数的详细解释可以参考这篇博客：[每天一个linux命令（7）：mv命令](http://www.cnblogs.com/peida/archive/2012/10/27/2743022.html)

**【26】查看进程**

基础使用方法（查找特定进程）：

ps aux|grep 你要查找的进程名

例如：

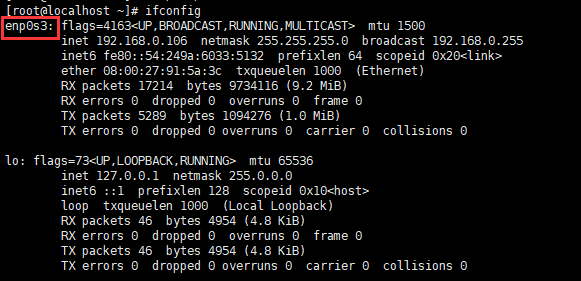
ps aux|grep nginx

详细版解释参照： [每天一个linux命令（41）：ps命令](http://www.cnblogs.com/peida/archive/2012/12/19/2824418.html)

【27】添加 IP别名

所谓IP别名，简单来说，就是在一块物理网卡上绑定多个IP地址，即，访问A IP和访问B IP实际上访问的都是同一台机子。

首先通过ifconfig查看当前网卡名、ip地址，netmask，broadcast



以上面为例，下面输入以下命令来添加一个新的ip地址（建议在原本基础上加1，并且不能和其他的ip地址重复，例如原本是192.168.0.106，新增的为192.168.0.107），而broadcast、netmask和之前保持一致。

格式：（临时添加的方法）

/sbin/ifconfig [网卡名]:1 [新的ip] broadcast [原本网卡的broadcast的值] netmask [原netmask值] up

/sbin/route add -host [新的ip] dev [网卡名]:1

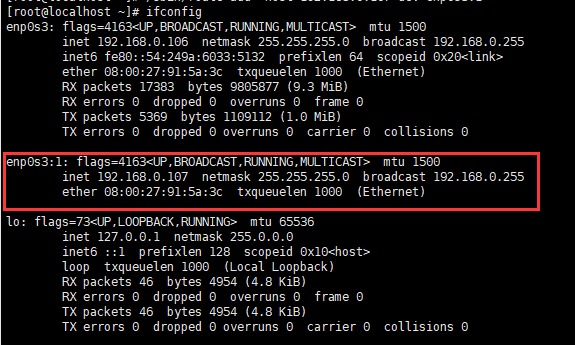
示例：

/sbin/ifconfig enp0s3:1 192.168.0.107 broadcast 192.168.0.255 netmask 255.255.255.0 up

/sbin/route add -host 192.168.0.107 dev enp0s3:1

如果添加第二个，那么就是网卡名后面的 :1 变为 :2 ，依次类推。

添加成功后，再次ifconfig查询如下



并且可以ping通192.168.0.107这个ip地址。

